IEICE TRANS. INF. & SYST., VOL.E99–D, NO.4 APRIL 2016

LETTER

Feature-Based On-Line Object Tracking Combining Both Keypoints and Quasi-Keypoints Matching

Quan MIAO[†a)], ***Nonmember***, **Chun ZHANG**[††b)], ***Student Member***, ***and*** **Long MENG**[†††c)], ***Nonmember***

SUMMARY This paper proposes a novel object tracking method via online boosting. The on-line boosting technique is combined with local features to treat tracking as a keypoint matching problem. First, We improve matching reliability by exploiting the statistical repeatability of local features. In addition, we propose 2D scale-rotation invariant quasi-keypoint matching to further improve matching efficiency. Benefiting from SURF feature's statistical repeatability and the complementary quasi-keypoint matching technique, we can easily find reliable matching pairs and thus perform accurate and stable tracking. Experimental results show that the proposed method achieves better performance compared with previously reported trackers.

key words: *object tracking, statistical repeatability, quasi-keypoint, on-line boosting*

1. Introduction

Object tracking has attracted much attention due to its wide variety of applications, from visual surveillance to human-computer interfaces. Robust tracking involves several challenging issues, including low image quality, appearance variations, illumination changes, cluttered backgrounds and occlusion.

Many different tracking methods have been proposed, from global template-based trackers [1]–[3] to local feature-based trackers [4]–[6]. The global trackers attempt to localize the object region using a bounding box and distinguish it from background. In contrast, feature-based trackers detect local features from the object region and establish feature correspondences using distinctive descriptions. Then the motion parameters are estimated based on the set of matching candidates.

One critical issue is the model degradation caused by inaccurate identification of the object region. Current model updating is built on previous object region [7], thus slight target locating errors accumulated in long-term tracking will cause the appearance model to be updated with a sub-optimal positive sample. Over time this can cause drift and degrade performance.

To overcome the above problems, this paper proposes a feature-based object tracking method by incorporating both keypoints and quasi-keypoint matching. As for keypoint matching, we exploit the statistical repeatability of local features. When a new frame arrives, only high confident classifiers will be used, which leads to adaptive tracking to large appearance change. However, in some cases keypoints detected in one image are not detected in another correspondingly, due to geometric or photometric transformations. Thus we propose 2D scale-rotation invariant quasi-keypoint matching to improve matching efficiency. Experimental results show the satisfying tracking performance of the proposed algorithm on several challenging video sequences.

2. Motion Model

In the reference frame (i.e., the keyframe), we extract local features within the object region and initialize object representation. The object representation contains sparse set of SURF points [9] $\{b_1, b_2, \ldots, b_M\}$ and their topological structure in the image domain. When a new frame arrives, we first detect its SURF points, and then perform keypoint and quasi-keypoint matching with the object representation. As for keypoint matching, we employ our previously proposed scheme [6] using on-line classifiers. The classifying mechanism mainly uses the boosting technique in [4]. The algorithm of quasi-keypoint matching is presented in Sect. 3. We are able to obtain target's new location after homography estimation by RANSAC. Finally, we update the model to make the tracker adaptive to subsequent changes.

3. Proposed Algorithm

3.1 Statistical Repeatability of Local Features

The SURF-based detector has been proven to outperform most previous detectors in terms of robustness and detection repeatability [8]. However, in some cases the appearance variation between the current object and the original model is significant, causing a certain part of feature correspondences to become a nonentity. If we still use the classifiers to find their matches, the incorrect matching candidates would result in invalid motion estimation. To overcome this problem, we exploit the statistical repeatability of SURF features, which can be illustrated in Fig. 1. Generally,

Manuscript received November 10, 2015.
Manuscript publicized January 21, 2016.
[†]The author is with the National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China.
[††]The author is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China.
[†††]The author is with Shandong Mingjia Technology Limited Company, Beijing 100084, China.
a) E-mail: miaoq@tsinghua.org.cn
b) E-mail: 13811989049@163.com
c) E-mail: dreammameng@gmail.com
DOI: 10.1587/transinf.2015EDL8232

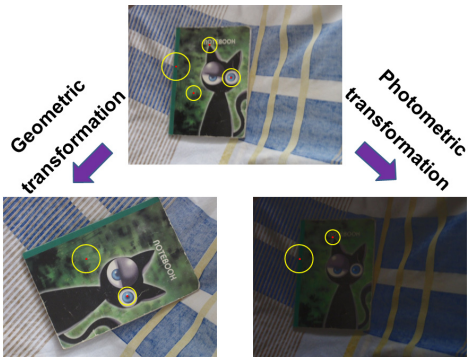

Fig. 1 Local features' repeatability change under different transformations.

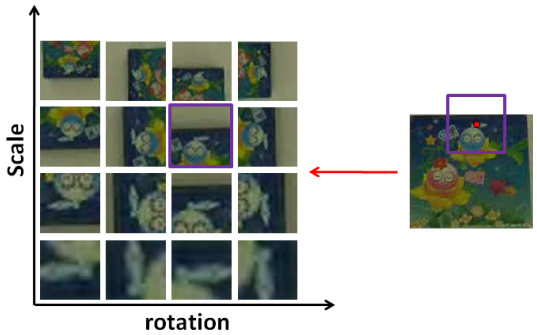

Fig. 2 Construction of the general 2D scale-rotation space for object keypoint of the model.

the SURF points with large gray contrast along horizontal or vertical direction are robust to photometric transformations. If the circle has a sharp center with smooth surrounding area, geometric transformation affects little.

In our scheme, we use detected keypoints $\{b_1, b_2, \ldots, b_M\}$ as the representation of the object model. During initialization of the object model, we build M classifiers $\{C_1, C_2, \ldots, C_M\}$, each corresponding to a keypoint b_m of the model. Given the keypoints set $\Upsilon = \{\gamma_1, \gamma_2, \ldots, \gamma_Q\}$ detected in the new frame, we employ the classifier C_m to compute the confidence measure of each γ_q, and record point φ_m by:

$$\varphi_m = \arg\max_{\gamma_q \in \Upsilon} C_m(\gamma_q). \tag{1}$$

To further develop the statistical repeatability of SURF features, we focus on those hat can best describe the current object change with the object model, by on-line determining a ranking of their confidence measure. For each φ_m, we compare its matching score $C_m(\varphi_m)$ with a threshold λ. If $C_m(\varphi_m)$ exceeds λ, we consider the classifier C_m shows good confidence of their adaption to current appearance variations and select φ_m as a reliable matching candidate. In case $C_m(\varphi_m)$ is below λ, we discard φ_m and convert to employ quasi-keypoint matching, which is illustrated in Sect. 3.2.

3.2 Quasi-Keypoint Matching

For each object keypoint of the model, we employ a $N \times N$ neighborhood from the reference image to capture its variation in the local appearance. Although the patch itself is fixed, we improve its property using scale space representation. The scale space is divided into octaves. Image patches in different octaves are centered around the corresponding keypoint with the same size but describe scale changes using image interpolation and down-sampling. Each octave represents a series of patches with rotation changes. The whole space starts from the practical patch around the detected object interest point. First we locate this patch in the correct position of the space based on the corresponding keypoint's actual scale and dominant orientation information (similar to the space representation in [9]), and then enrich its neighborhood. Figure 2 shows the process of the general 2D scale-rotation space construction for a certain interest point. The scaling factors for the four octaves are 1.0, 2.0, 4.0, 8.0 respectively. The degree intervals of dominant orientation for each octaves are (0, π/2), (π/2, π), (π, 3π/2), (3π/2, 2π), respectively.

In most practical situations, classifier-based matching in Sect. 3.1 is reliable to generate a good matching candidate. However, sometimes even the maximum confidence for a certain classifier is too small, especially when significant transformations occur. Thus it's highly possible that the point φ_m is a wrong correspondence and the true corresponding point has not been detected at all, which will cause false matching candidate. Right now we employ the corresponding 2D space and switch to "quasi-keypoint matching". Specifically, we sequentially use each patch of the space corresponding to point b_m as the template to find its most matched patch in the searching area of the current frame. Among the patches of the general 2D space, we record the one providing the best matching score and its position in the 2D space. The matched patch in the current frame is the so-called quasi-keypoint (location is the center of the patch), whose scale and dominant orientation can be obtained based on the recorded 2D position. Then the matching candidate φ_m changes into this quasi-keypoint and the matching score is updated.

With the help of the proposed quasi-keypoint matching, we are able to improve the detection repeatability and obtain more correct matching pairs. However, the template-based quasi-keypoint matching results in higher computation, compared to the classifier-based keypoint matching. Thus we restrict the searching area to a local region around the object region in the last frame. Also, sampling the local region with a certain sampling step can further improve computing efficiency without sacrificing the performance.

Similarly, the set of matching candidates $\Psi = \{\varphi_1, \varphi_2, \ldots, \varphi_M\}$ is established by using the same operation on Υ. Figure 3 shows an example of our combined matching. The quasi-keypoints are represented by rectangles. Such establishment of reliable matches is better suited for accurate motion estimation.

3.3 Model Updating

After identifying the current object region, we will update all the components. First we use the estimated motion parameter to perform a verification procedure over the suggested matches, obtaining a subset of correct matches (inliers). On the one hand, if the correct match comes from keypoint matching, we update the corresponding classifier and the 2D scale-rotation space representation, using the matched keypoint as positive sample. More details can be referred in [6]. On the other hand, we only update the 2D space in case the correct match belongs to quasi-keypoint matching. We apply no updates on false matching candidates. Finally, we record the current object location. When next frame arrives, we efficiently detect its keypoints within a local neighborhood covering the current object region.

No matter whether keypoint matching or quasi-keypoint matching each pair of correct match is from, we apply our 2D space representation updating scheme. We crop out the local patch Y around the positive sample in the current frame and locate it in the corresponding position in the 2D general space. For keypoint matching, the corresponding position is determined by the scale and dominant orientation of the matched keypoint. The corresponding position for quasi-keypoint matching is recorded by the patch of the 2D space providing the best matching score. Assume the id of the corresponding positive in the 2D space is i and the patch P_i already has L_i observations, we first update P_i by:

$$L_i = L_i + 1, \tag{2}$$

$$P_i = P_i + \frac{1}{L_i}(Y - P_i). \tag{3}$$

Equation (3) works out the weighted sum of the two patches, which is visualized using Fig. 4. Similarly, the other patches in this 2D space can be updated after a series of appropriate scale and rotation transformations on the local patch around the positive sample.

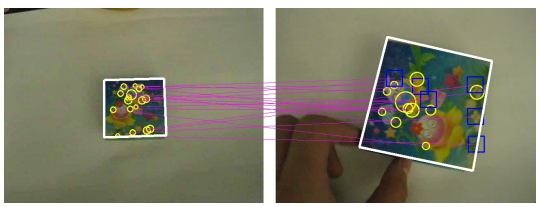

Fig. 3 Matching based on both keypoints and quasi-keypoints.

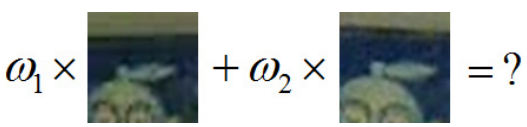

Fig. 4 The process of patch updating by weighted sum of the two patches.

4. Experimental Results

We carry the proposed algorithm on various videos with the size of 640 × 480 for experimental verification. These sequences are captured under complex appearance changes, containing scale, rotation and viewpoint transformations. For each keypoint, a classifier consists of 20 selectors selected from a global weak classifier pool holding 250 weak hypotheses. The general 2D scale-rotation space for each point contains totally 4×4 patches for quasi-keypoint matching and updating. The size of each patch is 40 × 40. After combined keypoint matching, we choose the best 40 matching candidates to perform RANSAC. If the number of inliers exceeds 10 (the percentage is above 25%), the object is tracked based on the estimated homography. To accelerate the whole tracking process, we implement the SURF-based keypoint detection using CUDA [10]. The platforms used are NVIDIA's GTX480 card and Intel E8400 3.0GHz CPU. The resulted CPU-GPU cooperative tracking system runs at a real-time speed of 20fps. All parameters are kept fixed for all the experiments.

For comparison, we implemented Grabner's tracker [5]. Meanwhile, our previous feature-based tracker [6] is used. Associated parameters and thresholds of these methods are fixed to be the same as we use in this paper. Figure 5 shows the performance of tracking a bottle under rapid appearance change. Obviously, Grabner's tracker fails once complex appearance occurs (e.g., the 53th frame), because the keypoints are detected using Harris corner very sensitive to scale and rotation changes. As for the previous feature-based tracker, drifting has seriously influenced the performance because tracking region is distorted (e.g., the 252th frame) as the tracking errors accumulated over frames, which leads to tracking failure in the 351th frame. In contrast, the proposed method successfully overcomes the drifting problem. The reason is two-fold. First, we focus on the relation between the reference frame and each incoming frame instead of successive frames. Second, our making use of statistical repeatability helps to improve local features' repeatability.

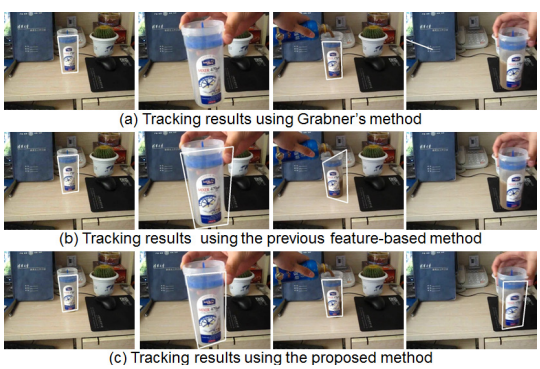

Fig. 5 Tracking a bottle under scale and rotation changes. From left to right column, the first, 53th, 252th and 351th frame.

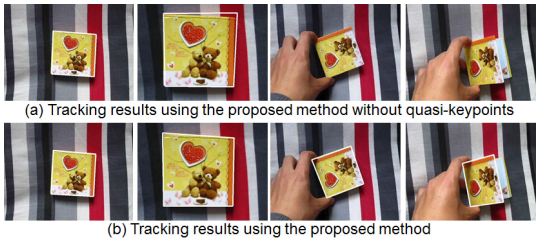

Fig. 6 Tracking a notebook under rapid scale and rotation change. From left to right column, the first, 63th, 158th and 348th frame.

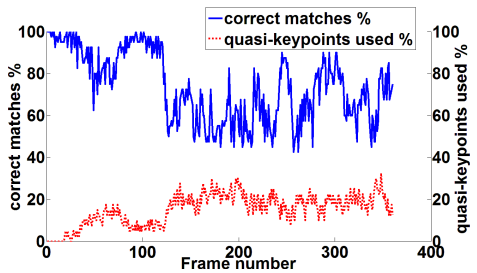

Fig. 7 Number of correct matches of the proposed algorithm and the percentage of quasi-keypoint used.

Next, we validate the superiority of using quasi-keypoints. For comparison, the proposed method without quasi-keypoints is added. As is shown in Fig. 6, the proposed method without quasi-keypoints fails once the object change becomes complex (e.g., the158th and 348th frame). In contrast, supplementing quasi-keypoints enhanced the total detection repeatability and thus ensured the matching efficiency, which is well suited for the affine changes.

To make a quantitative comparison, we consider the number of correct matches certificated by RANSAC for each frame using the proposed method. As the object changes are relatively small in the first few frames, the number of matches keeps a high ratio. Thus the use of quasi-keypoints is not necessary. However, when significant object changes occur, correct matches of keypoints may not be sufficient. Right now we turn to quasi-keypoints matching. Figure 7 simultaneously shows a comparison between the proposed method with quasi-keypoints and that without quasi-keypoints. Take the 158th frame for example, large appearance changes occur. Tracking will fail (the percentage of correct matches is below 25%) without quasi-keypoints. More quasi-keypoints (38%) are employed for matching and the formed matching candidates (77%) become reliable again. Thus the keypoint matching and the quasi-keypoint matching complement one another and the tracking stability can be preserved.

5. Conclusion

This paper presents an on-line object tracking algorithm by incorporating both keypoints and quasi-keypoint matching. In contrast to existing approaches, we exploit the statistical repeatability of local features and therefore the most reliable matching candidates are selected using classifiers. In addition, the construction of 2D scale-rotation invariant quasi-keypoint further complements matching. Experimental results verify that our approach outperforms the state-of-art tracker to achieve stable and robust tracking.

References

[1] S. Avidan, "Ensemble tracking," CVPR, pp.494–501, 2005.
[2] V. Mahadevan and N. Vasconcelos, "Saliency-based discriminant tracking," CVPR, pp.1007–1013, 2009.
[3] A. Yao, X. Lin, G. Wang, S. Yu, "A compact association of particle filtering and kernel based object tracking," Pattern Recognition, vol.45, no.7, pp.2584–2597, 2012.
[4] H. Grabner and H. Bischof, "On-line boosting and vision," CVPR, pp.260–267, 2006.
[5] M. Grabner, H. Grabner, and H. Bischof, "Learning features for tracking," CVPR, pp.1–8, 2007.
[6] Q. Miao, G. Wang, C. Shi, X. Lin, and Z. Ruan, "A new framework for on-line object tracking based on surf," Pattern Recognition Letters, vol.32, no.10-12, pp.1564–1571, 2011.
[7] Q. Miao, G. Wang, X. Lin, Y. Wang, C. Shi, and C. Liao, "Scale and rotation invariant feature-based object tracking via modified on-line boosting," ICIP, pp.3929–3932, 2010.
[8] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," in Foundations and Trends in Computer Graphics and Vision, vol.3, no.3, pp.177–280, 2008.
[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," Computer Vision and Image Understanding, vol.110, no.3, pp.346–359, 2008.
[10] NVIDIA, NVIDIA CUDA programming guide version 2.3, NVIDIA, 2009.